

**What is claimed is:**

1. A computer system, comprising:
  - a memory space having a number of memory locations;
  - an operating system located within a system space, the system space corresponding to a first subset of the number of memory locations of the memory space;
  - a software module located within a user space, the user space corresponding to a second subset of the number of memory locations of the memory space;
  - a plurality of operating system data structures located in the system space;
  - a system page located within the system space and corresponding to a portion of the first subset of the number of memory locations, the system page including a subset of the plurality of operating system data structures; and
  - a function located within the software module;wherein the function may not be linked to the first subset of the number of memory locations except for the subset of the plurality of operating system data structures.
2. The system of claim 1, further comprising:
  - a task, the task assigned to execute the function and having a memory map indicating access to the user space and access to the system page.
3. The system of claim 2, wherein the memory map includes access indicators of read-only access to the system page.
4. The system of claim 2, wherein the portion of the first subset of the number of memory locations that corresponds to the system page are associated with access indicators of read-only access by user tasks.
5. The system of claim 1, wherein the portion of the first subset of the number of memory locations that corresponds to the system page is contiguous.

6. The system of claim 1, wherein the portion of the first subset of the number of memory locations that corresponds to the system page is page-aligned.
7. The system of claim 1, wherein the subset of the plurality of operating system data structures includes an identifier of a currently executing task.
8. The system of claim 1, wherein the subset of the plurality of operating system data structures includes a timer.
9. The system of claim 1, wherein the subset of the plurality of operating system data structures includes a counter indicating a current level of interrupt nesting.
10. The system of claim 1, wherein the subset of the plurality of operating system data structures includes a pointer to a task memory block.
11. The system of claim 1, wherein the operating system is a real-time operating system.
12. A method, comprising the steps of:
  - creating a task, the task assigned to execute at least one function;
  - assigning a memory access map to the task including indications of access to a portion of a memory space allocated to the at least one function;
  - setting access indicators of read-only access for memory locations of a system page including operating system data structures, the system page located within a system memory space;
  - including in the memory access map indications of access to the memory locations of the system page; and
  - allowing a read memory access by the task to the memory locations of the system page.

13. The method of claim 12, further comprising:  
executing an exception handling routine when the task attempts a write memory access to the memory locations of the system page.
14. The method of claim 12, further comprising:  
signaling an error condition indicating that the write memory access was attempted.
15. The method of claim 12, wherein the operating system data structures include an identifier of a currently executing task.
16. The method of claim 12, wherein the operating system data structures include a timer.
17. The method of claim 12, wherein the operating system data structures include a counter indicating a current level of interrupt nesting.
18. The method of claim 12, wherein the allowing of the read memory access by the task to the memory locations of the system page is performed by a memory management unit.
19. The method of claim 12, wherein the setting of access indicators of read-only access for memory locations of the system page occurs prior to the creation of the task.
20. The method of claim 19, wherein the access indicators are associated with the memory locations of the system page.
21. The method of claim 12, wherein the memory access map includes a range of memory addresses.

22. The method of claim 12, wherein the memory access map includes a bit map of memory pages.

23. A method, comprising:

retrieving a software module having a symbol reference, the symbol reference used by an instruction;

resolving the symbol reference, including:

searching an operating system symbol table for a symbol entry corresponding to the symbol reference,

checking the symbol entry corresponding to the symbol reference for an indication that the symbol corresponds to a system page data structure,

obtaining a symbol value for the symbol from the symbol entry when the indication is present; and

inserting the symbol value into the instruction.

24. The method of claim 23, further comprising:

leaving the symbol reference unresolved when the symbol entry corresponding to the symbol reference does not include the indication that the symbol corresponds to a system page data structure.

25. The method of claim 23, further comprising:

loading the software module into a memory space.

26. The method of claim 23, further comprising:

generating an error message when the symbol entry corresponding to the symbol reference does not include the indication that the symbol corresponds to a system page data structure.

27. The method of claim 23, wherein the system page data structure is part of a system page located within a system memory space.